Instructor Introduction and Class Logistics

CS 10A – WELCOME! (ONLINE EDITION)

Instructor Introduction

- Allen Zhao (azhao@santarosa.edu)
- Hardware Design Engineer at Keysight Technologies
- University of California, San Diego
- B.S. NanoEngineering (2014)
- M.S. Electrical and Computer Engineering (2016)
- Teaching the night class at SRJC since SP19
- Primary languages: C, C#, Java
- Amateur in eSports and astronomy

Class Syllabus

- General introduction to programming and computer science topics
- Development of coding thought processes and habits
- Language will be C++ (strong overlap with C)
- How to use a Unix Terminal or industry standard IDE
- Lecture to learn new concepts and theories
- Lab to practice applications
- Concepts: IOs, Logic, Control, Functions, Arrays (and then some)
- Other concepts in math and physics as necessary

Class Expectations

- Math level must be, at minimum, Algebra I or equivalent.
- Recommended math level is Algebra II or equivalent (MATH 155 at SRJC).
- Basic computer skills and willingness to apply and develop critical thinking, logic, and pattern recognition skills.
- Your own laptop/computer for convenience.
- All course content will be made available on Canvas.
- You are responsible for your own learning. I won’t be taking attendance. Assignments are used to track participation.
- This is a very homework intensive class! Programming is best learned through continuous practice rather than through exams.

Basic Computer Skills

- The following criteria define the “basic computer skills” expected of all students entering this class.
  - Navigating folders/directories with Finder/Windows Explorer
  - Organizing/creating new file and folders
  - Finding files without using the search tool
  - Opening and utilizing the right click menu
  - Downloading files and installing programs (without an app store)
  - Accessing Settings and/or Control Panel
  - Transferring files between external memory storage and devices

Lecture Expectations

- If you’re not actively speaking during class, mute your microphone. It cuts down on background noise.
- Lecture will have at least one short break every hour.
- You’re free to leave whenever without notice.
- Disable your camera. As much I appreciate seeing your faces, there’s not enough internet bandwidth to support a smooth live feed from everyone simultaneously. Audio quality takes priority.
- Lectures tend to be short; only occasionally will we need all three hours to get through the topics of the week.
- Stick around after lecture if you have questions for me. I will log off for the day only after all students leave the Zoom call. Afterwards, I will not return for the rest of the night, so any questions thought of afterwards should be sent to my email (either my normal email or through Canvas is fine).
- All lectures will be recorded and linked to on Canvas.

Lab Expectations

- Apart from the first week, lab is optional to attend. Unlike lecture, this is basically a Q&A session where I hover around online for all three hours. This is your chance to actively work while I'm around to give you feedback and get lab assignments checked off for full credit.
- Ask questions! If you're having problems getting your program to compile, ask me for assistance. Some errors are beyond just the code.
- Though not required, I recommend you partner up with someone to work with for the semester. Use the classroom's Discord server to request or respond to requests for class partners.
- Office Hours will use the same Zoom link as the Lab classroom, every Tues. 6-7:30pm.
- Save and back up your work regularly! I recommend using a cloud storage like Google Drive or a USB thumb drive to back up your work.
- Students joining the Lab/Office Hours Zoom meeting will first be dropped into a waiting room while I address your questions one to one on a first come first serve basis. I will call in the next student after the previous student logs finishes asking questions. Please be patient as the student ahead of you asks me their questions. One-on-one appointments to privately address sensitive topics like extensions or grades are also available upon request, even in the middle of Lab hours.
  - For peer-to-peer classroom collaboration, hop onto the Discord server during lab hours. Help your classmates!
- You do NOT need to be present at 6pm on typical Lab days. Feel free to drop in any time between 6pm and 9pm to ask your questions. This prevents a huge queue from forming at 6pm.

Class Rubric

- Exams: 30%
  - Midterm: 15%
  - Final: 15%
  - Both exams will be curved
- Programming Assignments: 60%
  - Homework: 30%
  - Lab Assignments: 30%
- Worksheets: 10%
- Class will be curved if class average falls below 80%.

Assignment Policy

- Every week will have Lab and Homework assignments given, excluding holidays.
  - Labs are assigned Monday and due on Wednesday (can be checked off).
  - Homework is assigned Wednesday and due the following Wednesday.
- Worksheets are given on some Mondays, generally given two weeks to complete.
  - Exams mostly cover Worksheet topics but may also include programming problems.
- All assignments have a 2-day late period submission window. For each day late, 10% of the max is deducted from your score. Solutions are posted afterwards.
- For all assignments, 11:59pm is the Canvas submission cutoff on the due date.
- Assignments scoring less than 50% can be resubmitted any time after the late deadline to me via email to achieve a max score of 50%. This also applies to assignments you have missed altogether. Copy the solutions if you want. My only concern is that you are actively LEARNING. Cheating is self-regulating in this field.

Assignment Policy

- Leveling the playing field: you are NEVER allowed to use any topic or concept we haven't yet covered in class on any of my programming assignments.
  - Ensures veterans have the same challenge as rookies.
  - Ensures rookies master the basics before using shortcuts.
  - Learn to SOLVE problems, not Google for a solution.
- Most assignments will come with a subset of additional requirements you'll need to meet to get full credit.
  - There's more to programming than just getting something to work.
- Certain techniques, even as I teach them, will be forbidden to use at all times. This will be clarified as we progress.

Tips for Success

- Engagement matters! Talk to me, talk to your classmates, find a tutor.
- Stay healthy, physically and mentally. Drink water. Get plenty of sleep.
  - Turn down notifications for social media apps. Cut the digital noise.
- You must think like a computer, not like a human, to become a good programmer. Know the difference. Internalize it. Make use of both.
- CS is hard to learn, but it doesn't have to be complicated. Don't overthink the problems. The answers are often simpler and shorter than anticipated.
- Learn to interpret error messages from your compiler. They're very handy.
- Strong pattern recognition skills and abstract thinking are the most critical skills for becoming a great programmer.
  - An intuition for numbers also helps a lot. (e.g., How much is 1%, really?)
- "Take chances, make mistakes, and get messy!" – Ms. Frizzle. Engineering is all about being willing to learn from the inevitable failures.

Tips for Success

- Google may hurt you more than help you. Stick to the class content.
  - If you're not careful, the lack of an information filter will only confuse you further.
- Do not rely on your computer or calculator to figure things out for you.
  - You must learn to be better than the computer. It is your assistant, not your master.
- A lot of hints to assignments can be found in the lecture slides.
- Every week's new content builds upon the previous. Study solutions to past assignments to get caught up. Stay on top of things! This class gets busy.
- New assignment posted? Look at it right away. Nothing says you must start right away, but even a glance can help get your brain thinking early.
- Read all directions carefully! Aside from my additional requirements, I often leave hints that can help you navigate the problem more easily.
- I can teach more than just programming.
  - Physics? Math? Career advice? Resume critiquing? Can't even? I'm all ears.

Writing Better Emails

- Emails are not DMs! Practice professionalism and speed up your productivity by writing better emails to me when you need to ask questions outside of class hours.
- Tip #1: When asking about technical issues, provide a screenshot of the error in question.
- Tip #2: Attach your code file if you have a question about your code. Specify the line number and what you've already tried to rectify the error.
- Tip #3: Emails are long-form messaging. Don't hesitate to write more if it helps with describing the issue you're having.

Basic Terminology in CS

- Block – a term used to generally indicate a section of a program
- Compile – compiling a program means getting the computer to translate your code into something it can actually execute
- Compiler – a program that compiles programs
- Console – the basic computer command terminal that's usually a black window with green/white text
- Environment – refers to your digital workspace, i.e. operating system, software
- IDE – Integrated Development Environment, generally refers to any kind of software that allows a user to create programs
- IO – sometimes written as I/O, short for input/output
- Runtime – refers to when a program is actively executing or running
- Script – a term generally used to indicate any one program file
- Syntax – a term that refers to programming language grammar, i.e. use of proper symbols, capitalization, format, etc. May vary with language

Programming - Oversimplified

- Step 1 – Create the program file (use a text editor)
  - Files must be of .cpp extension (indicates C++ language file)
- Step 2 – Save and compile the program file
  - Requires you to install a compiler (see Getting Started Modules)
  - Compiling the file will generate an executable file (.exe or .out)
  - The compiling process will vary depending on your installed tool
- Step 3 – Run the generated executable file (.exe or .out)
  - .exe files are generated for Windows users
  - .out files are generated for MacOS and Linux users